LA-UR-03-7607

*Approved for public release;
distribution is unlimited.*

Title:

MSTK (Mesh Toolkit): Flexible Framework for Representing and Manipulating General Unstructured Meshes

Author(s):

Rao Garimella (rao@lanl.gov)
T-7, LANL

Submitted to:

Internal Presentation

Los Alamos
NATIONAL LABORATORY

MSTK (Mesh Toolkit)
Flexible Framework for Representing and Manipulating General Unstructured Meshes

Rao Garimella

T-7, Mathematical Modeling and Analysis,
Los Alamos National Laboratory,
Los Alamos, New Mexico 87544.

Introduction

- Many applications based on unstructured meshes
  - — Finite element/volume methods, computer graphics
- Lots of common ground in
  - ◇ the types of mesh data stored by these applications
  - ◇ the forms in which this data is accessed
- However, very few tools that address these common needs
- Each application usually designs, implements and maintains its own mesh data structure based on specific needs
- Duplication of work, waste of time and money

Mesh Toolkit/Database/API/Framework

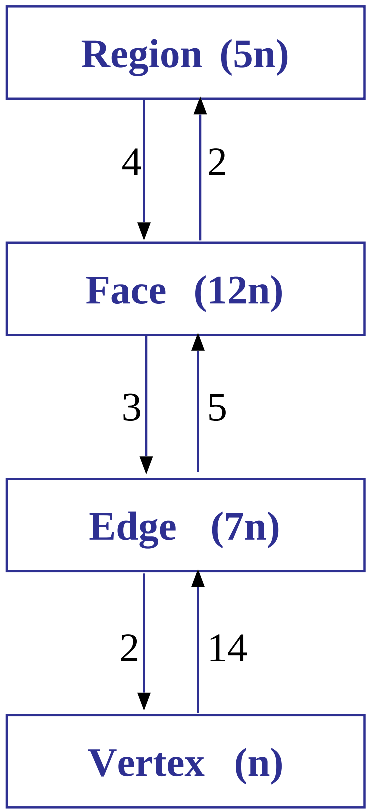
Need software infrastructure that will provide commonly used functionality for storing, retrieving and manipulating unstructured mesh information

Mesh Representations - Common Ideas

- **Most applications use concept of mesh ENTITIES**
  - ◇ 0D - vertices/nodes, 1D - edges, 2D - faces, 3D - regions
  - ◇ Explicit or implicit
- **Most applications use concept of ADJACENCY**
  - ◇ Adjacency: connectivity of entities of different dimensions
  - ◇ Examples: nodes of a element, regions around an edge
- **Each application uses a particular mesh representation**
  - ◇ specific combination of mesh entities and ajdacency relationships that are explicitly stored

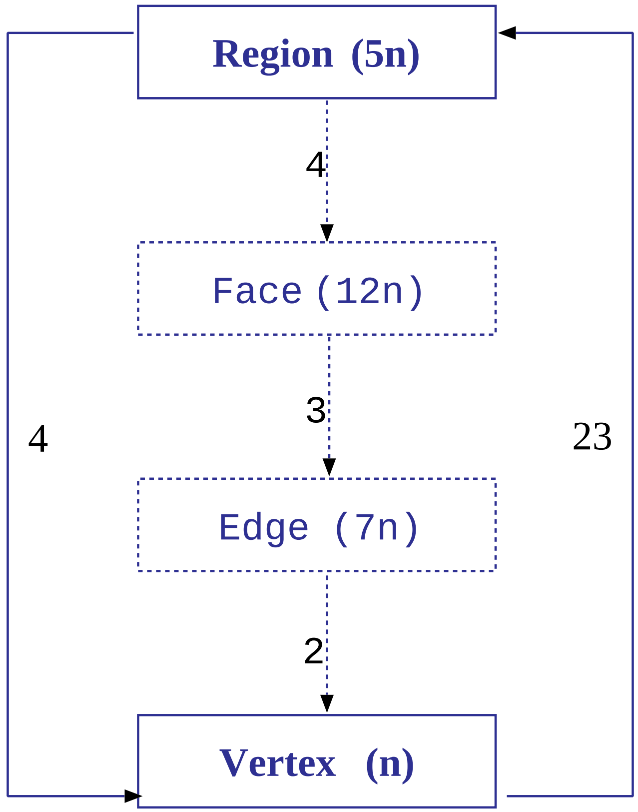
Full Mesh Representations

All entity types upto the dimension of the mesh are explicitly represented



Reduced Mesh Representations

Some intermediate entity types may not be represented explicitly



Cost of Using a Mesh Representation

- **Storage cost**
  - ◇ **memory usage of stored entities and adjacencies**

- **Computational cost**
  - ◇ **primarily from adjacency retrieval in static meshes**
  - ◇ **minimal for stored adjacency**
  - ◇ **substantial for derived adjacency**

Overall Computational Cost

- Op Count (OC) - cost of a mesh operation (e.g., adjacency retrieval)
- Relative Call Frequency (RCF) - Relative Number of times an operation is called in an application
- Overall computational cost = $\boxed{\sum_{ops} (OC)_{op} \times (RCF)_{op}}$

Need for Flexible Mesh Representations

- Toolkits usually choose one general representation to serve needs of wide range of applications
- But, RCFs of mesh operators vary among applications
- So, overall cost of using representation different for each application
- Therefore, one mesh representation not the most efficient for all applications
- This is one reason why applications avoid general toolkits

Flexible Mesh Toolkits

Need toolkit that supports different mesh representations to suit needs of different applications

However, interface for accessing mesh info must be uniform for all representations

Allows applications choose different representations without rewriting any code.

Ref: Remacle 1999, Garimella 2002

MSTK - Mesh Toolkit

- **MSTK - toolkit for low-level creation, modification, storage and retrieval of unstructured mesh data**
- **Hides details of data structures from applications**
- **Functional interface (API) for interacting with mesh representation**
- **Applications can concentrate on their own tasks**

MSTK is not a mesh generator but it can be used to write one!

MSTK - Multiple Mesh Representations

- MSTK allows applications to choose from multiple mesh representations
- May not find perfect representation for application
- Hopefully, adequately suitable representation can be found
- Can be expanded to support new representations

- MSTK also allows dynamic switching between representations
- Different representations for different algorithms in one application

Related Work

- **Algorithm Oriented Mesh Database (AOMD),** ***Rensselaer***
- **MDB, the Sandia Mesh DataBase Component,** ***Sandia Nat. Labs***
- **PMO, Parallel Mesh Object,** ***Terascale LLC***
- **GrAL, Grid Algorithms Library,** ***NEC C&C Res. Labs.***
- **TSTT Data Model,** ***SciDAC***
- **MSTKLA, R. Garimella,** ***EES5/LANL*** **(not maintained)**
- **MeshSim,** ***Simmetric Corp.***

Ref: http://endo.sandia.gov/cubit/mdb.htm

MSTK - Mesh, Mesh Entities and Entity Sets

- **MSTK supports multiple meshes simultaneously**
- **Each mesh contains mesh entities**
- **Entities: vertices (0D), edges (1D), faces (2D), regions (3D)**
- **Meshes and Mesh Entities are like C++ objects**
- **Applications cannot access any object's data structure directly**
- **Objects are referenced only by a generic pointer**

MSTK - Functional Interface

- **All interaction with objects is through operators (function calls)**
- **Operators exist for creation, querying, modification and deletion of mesh, mesh entities**
- **Even for reduced representations, applications can work as though all types of entities exist**
- **When necessary, MSTK will construct virtual entities**
- **Examples:** MESH_Next_Edge, MESH_Num_Faces, MV_Coords, ME_New, MF_Set_Edges, MR_ID

MSTK - Functional Interface (Contd.)

- **Operators exist for querying of multi-level adjacencies (connectivity information)**
- **Examples:** MV_Faces, MR_Edges, ME_Regions, ME_Vertex
- **Adjacency queries typically return entity sets**
- **Externally entity sets are also objects**
- **Interaction with entity sets is by operators only**
- **Examples:** Set_Num_Entries, Set_Next_Entry, Set_Delete

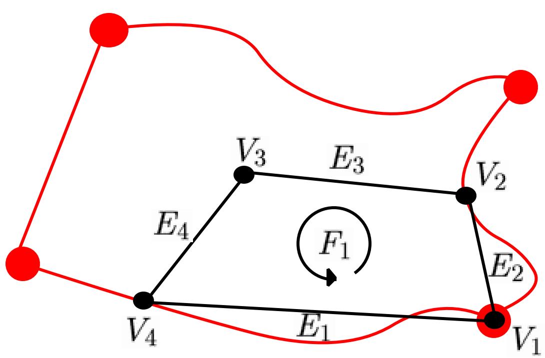
MSTK - Higher level operators

- **Topological: Edge swap, Join two faces, Collapse an edge**
- **Geometric: Vertex Coordinates of mesh face, mesh region**
- **Combined: Check if edge swap will result in "invalid" mesh**
- **Functionality being added as and when required**

- **Additional geometric functions being accumulated in a Computational Geometry library independent of MSTK**
- **Examples: volume of region, normal of triangle, line-line intersection, distance from point to line**

Relationship of mesh to geometric model

- Every mesh is discrete representation of a geometric model
- Classification: Relationship of a mesh entity to specific geometric model entity
- A mesh entity is partial or complete discretization of geometric model entity it is classified on



MSTK - Classification information

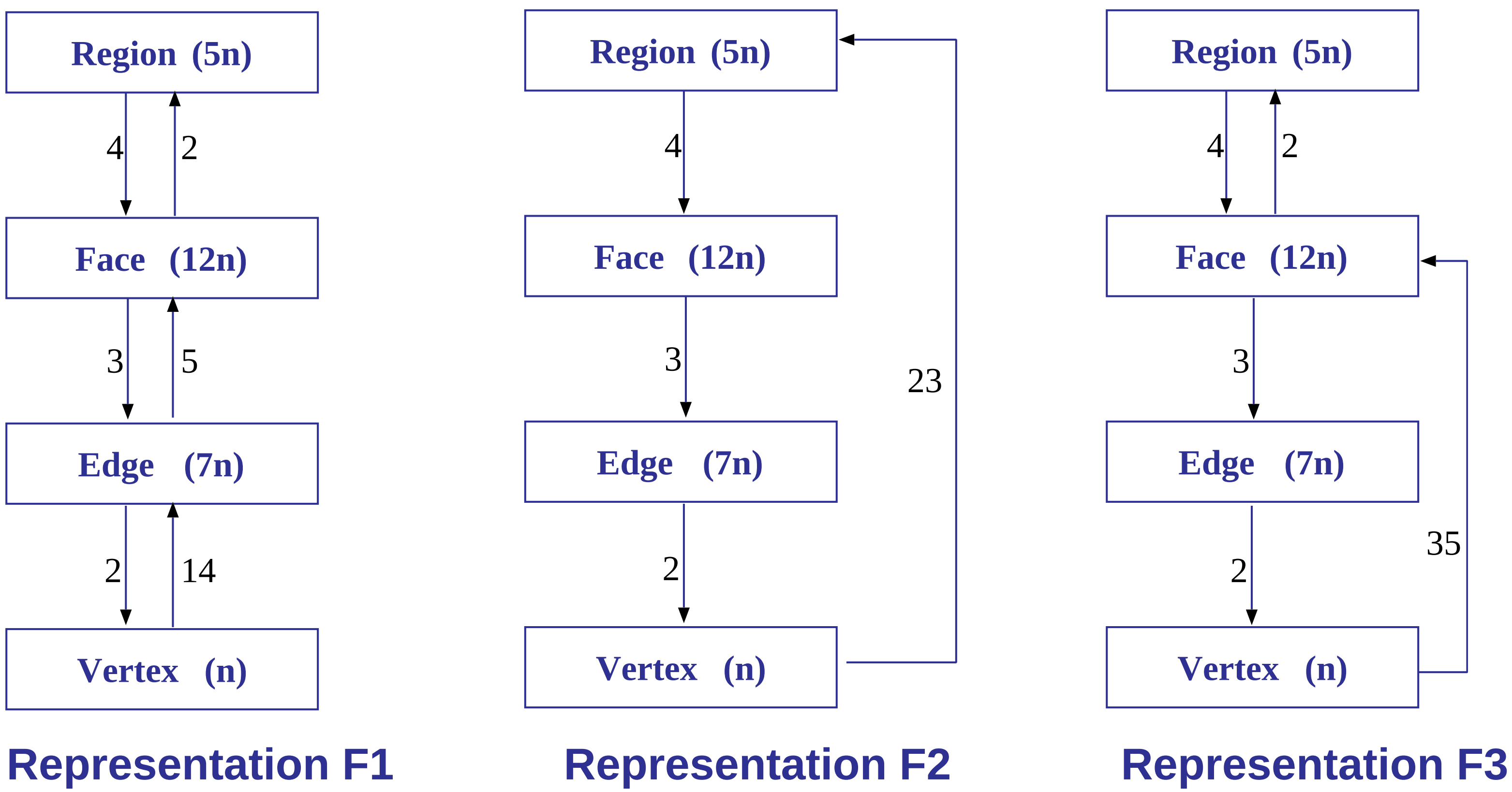
- MSTK allows full, partial or no classification information for each mesh entity
  - Type of geometric model entity
    - ◇ vertex, edge, face, region
    - ◇ useful for constraints, boundary conditions
  - ID of geometric model entity
    - ◇ useful for distinguishing material regions
  - Handle (pointer) to entity in geometric model
    - ◇ useful for extracting topological and geometric details from geometric modeler

Selection of Mesh Representations

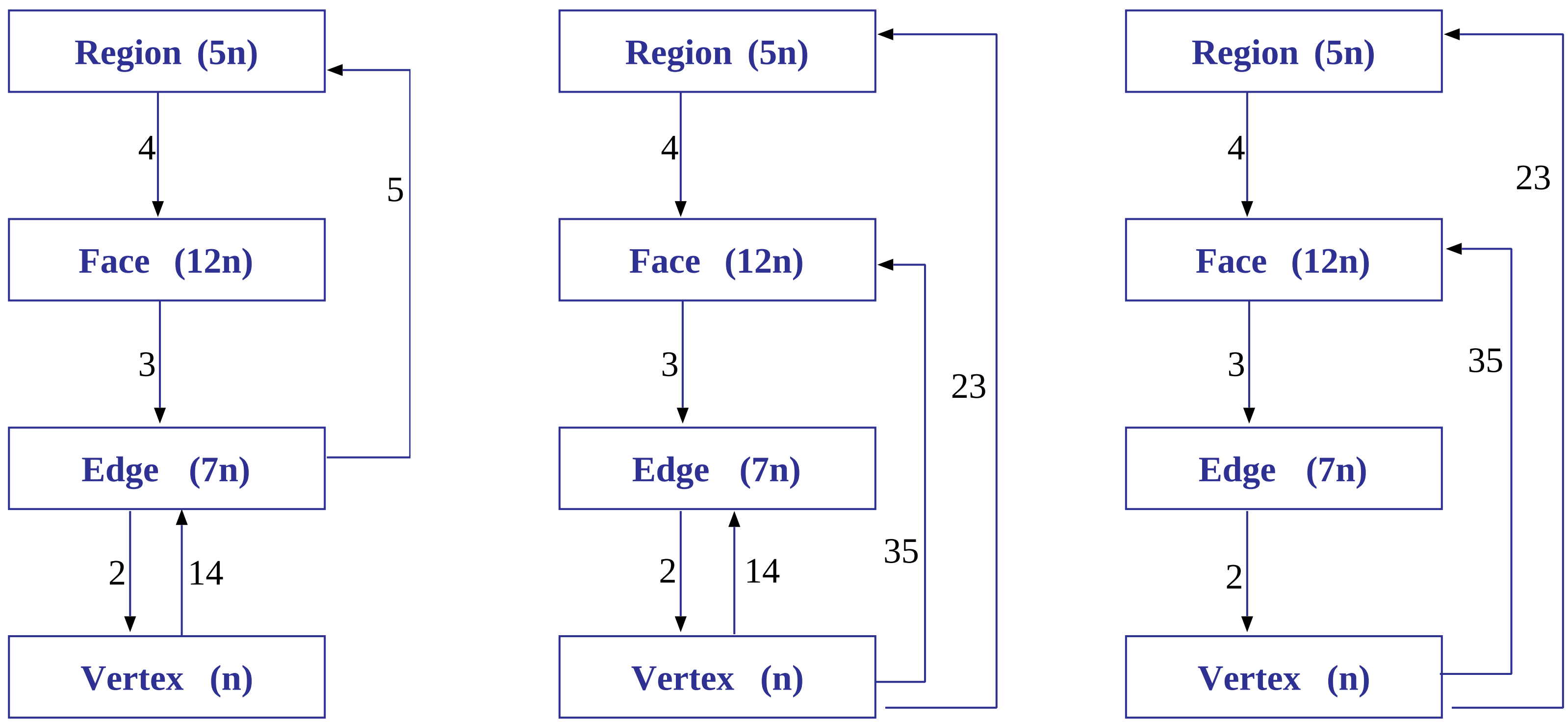
Which mesh representation should I use for my application?

- **Does it represent the mesh adequately?**
  - ◇ **cannot represent polyhedra by element/node data alone**
- **What is the cost to my application?**
  - ◇ **Account for Memory Usage and Computational Efficiency**
  - ◇ **IDEAL: minimize memory and computational cost**
  - ◇ **REALITY: compromise between memory, computational cost**

Full Mesh Representations - Examples

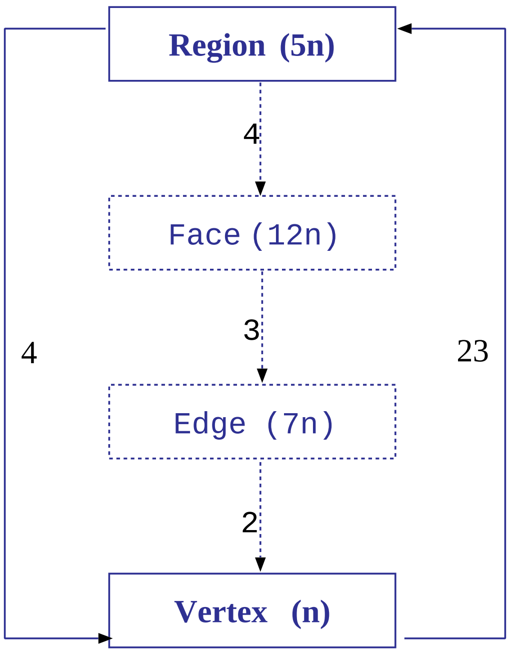


Full Mesh Representations (contd.)

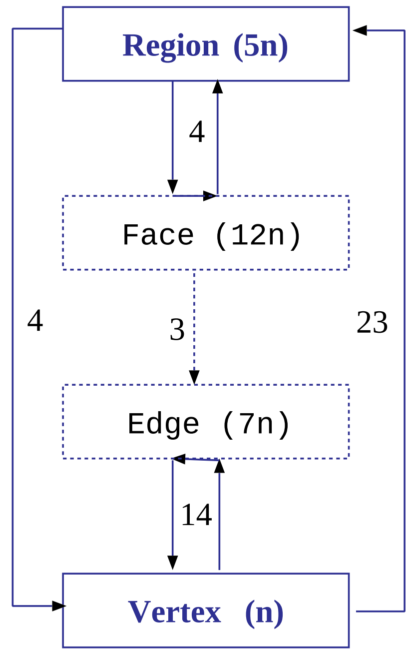


Representation F4 Representation F5 Representation F6

Reduced Mesh Representations - Examples

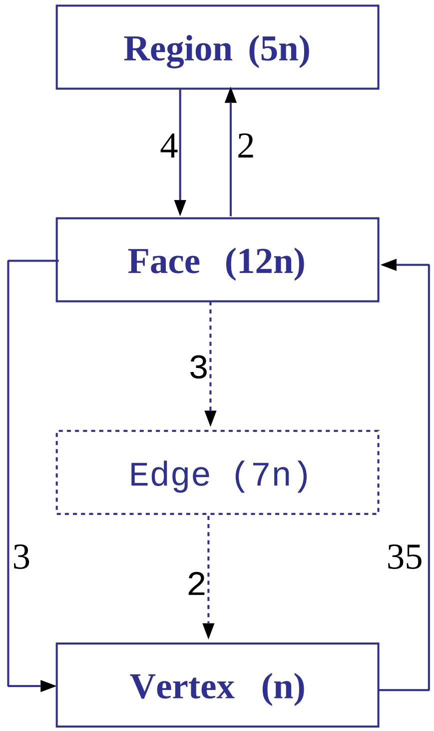


Representation R1

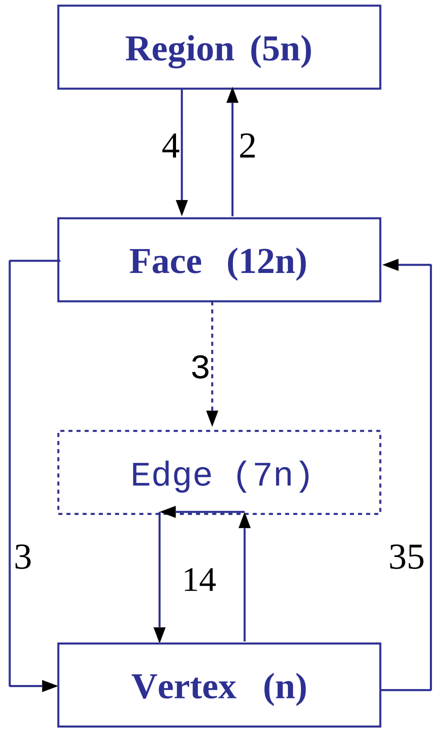


Representation R2

Reduced Mesh Representations (contd.)

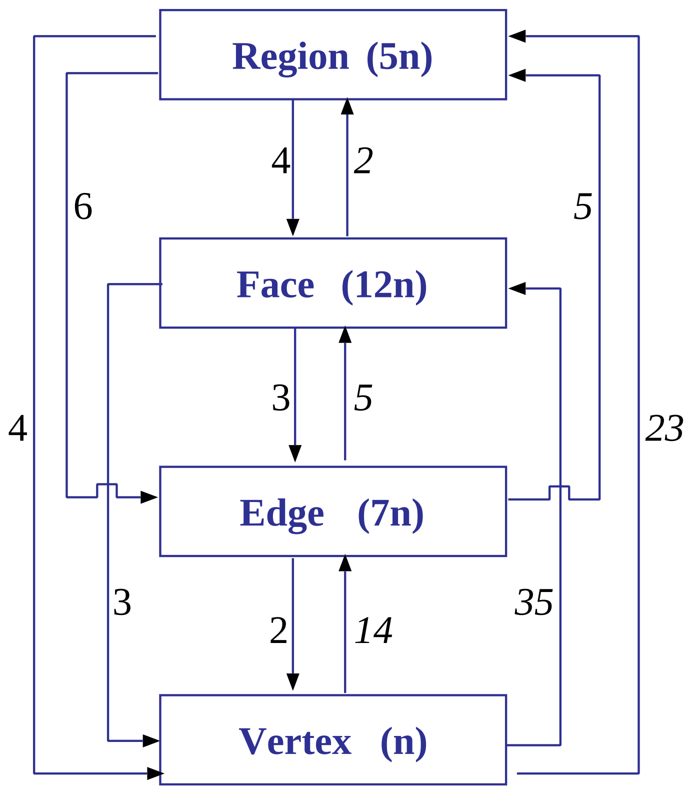


Representation R3



Representation R4

Memory Usage of Mesh Representations



- **Tetrahedral mesh example**
- **Assume memory usage of each entity is E, each connection is C**
- **Number of mesh vertices/nodes is n**
- **Memory usage of entities is** $(5n + 12n + 7n + n)E = \mathbf{25}nE$
- **Memory usage of connections is** $((5n)(4+6+4) + (12n)(2+3+3) + (7n)(5+5+2) + (n)(14+35+23))C = \mathbf{322}nC$
- **Total memory usage is** $(25E + 322C)n$

Ref: ***M.W. Beall, et.al., 1997***

Mesh Representation Rankings and Measures

	R2	F4	R4	R3	F5	F2	F6	F3	F1	R1
R_s (Mem. rank)	2	6	5	3	9	4	7	8	10	1
R_a (Comp. rank)	5	2	4	8	3	9	7	6	1	10
$\eta = \sqrt{R_s^2 + R_a^2}$	5.39	6.32	6.40	8.54	9.49	9.85	9.90	10.00	10.05	10.05
R (Overall rank)	1	2	3	4	5	6	7	8	9	9

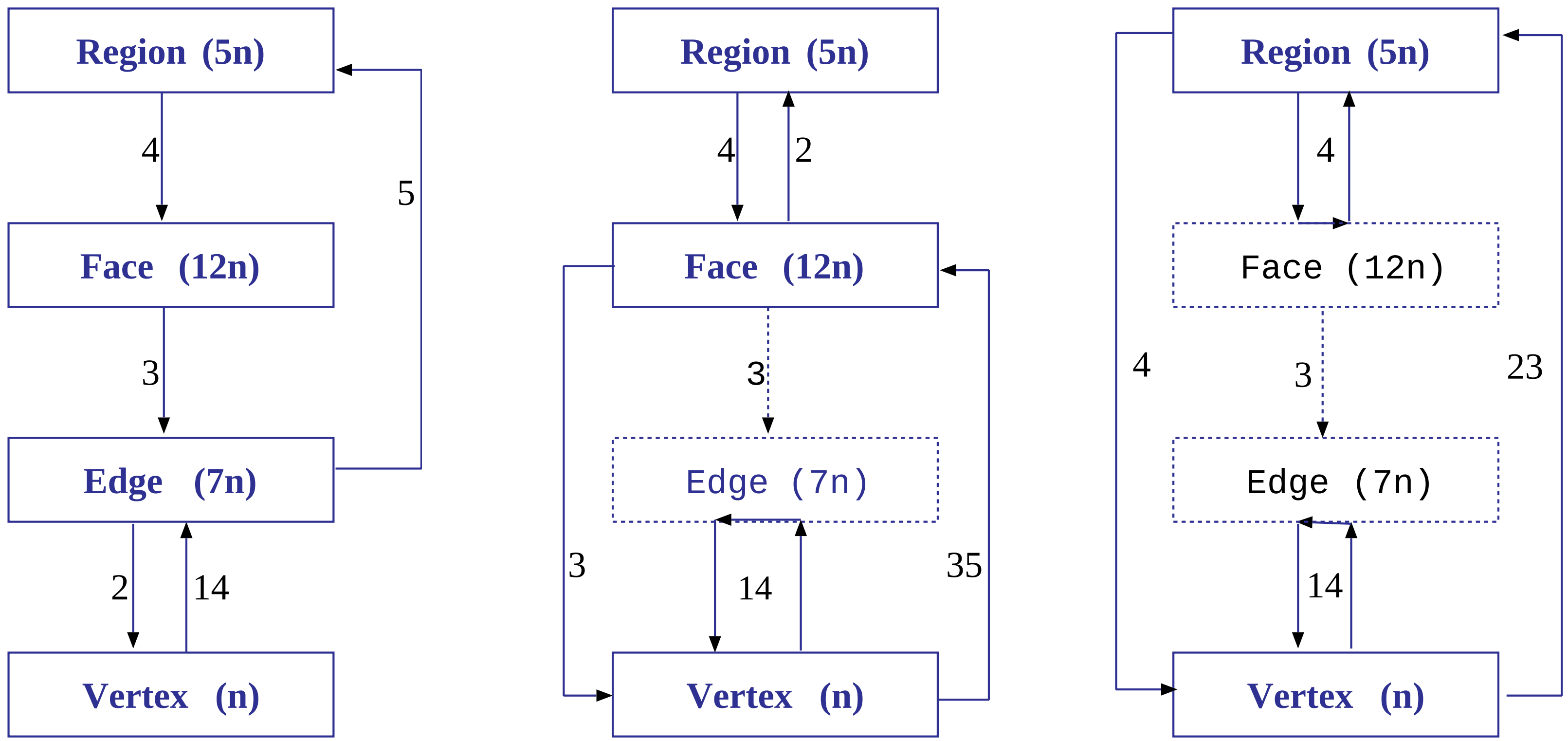
- η - cost index for minimizing computational and storage costs
- Storage rank obtained by assuming $E = 5C$

  Note: $E >> C$ usually and the rank is unchanged for $E > 3.15C$
- RCFs obtained by running large number of tests with mesh generator from SCOREC, RPI

Discussion of Mesh Representation Rankings

- R2, F4, R4 - good compromise between storage and computational efficiency
- Similar results for hexahedral meshes
- Top performers same in limited calculations for higher order finite element analysis of acoustic problems (*S. Dey, 2000*)
- For Relative Call Frequencies used here:
  - ◇ F4 is a good choice if a full data structure is necessary
  - ◇ R4 is a comparable choice if edges can be virtual
  - ◇ R2 is the best choice if both faces and edges can be virtual

"Good" Mesh Data Structures for Chosen Application (Revisited)



Representation F4 Representation R4 Representation R2

Mesh Representations in MSTK

- MSTK has implementation of representations F1, F4, R1, R2, R4
- F1 is a popular heirarchical representation and is fast
- R1 is popular element-node representation and is the leanest
- F1, F4 have been tested with triangular and tetrahedral meshes
- Dynamic switching of representations not yet in place
- Can only select the initial representation type for a mesh

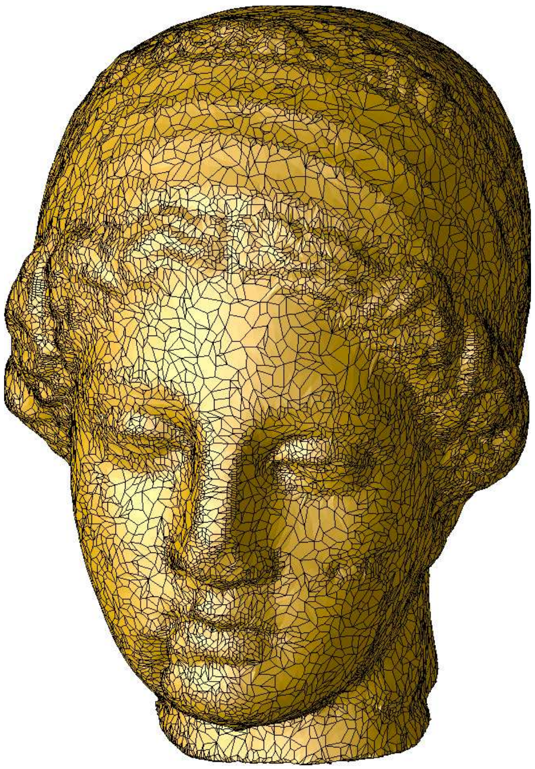
MSTK - Software specifics

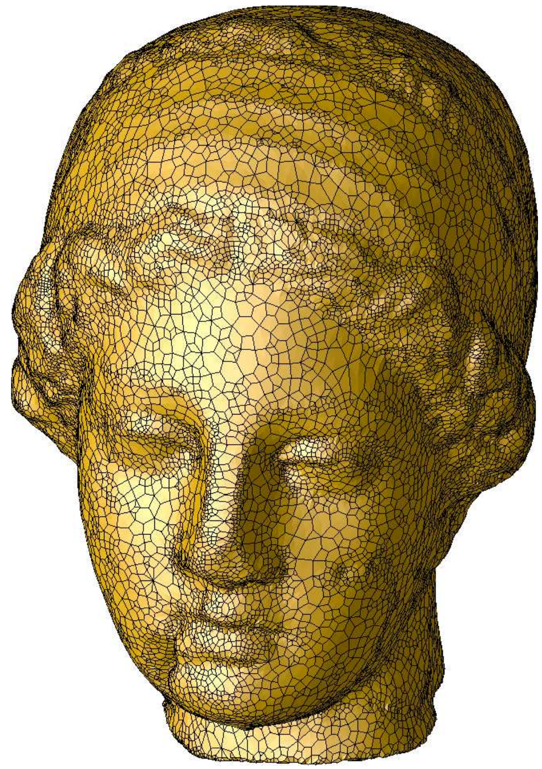
- Software coded in C for efficiency
- Data hiding achieved using only C functionality
- Internally, all objects are structures
- Externally, objects typedefed as equivalents of $(\mathrm{void}\ *)$
- Jump tables to invoke correct functions for each representation
- Code compiled on Intel-Linux using gcc with ANSI compliance

MSTK - Applications

- MSTK being used for number of real applications
- Optimization and Untangling of meshes, 3D conservative remapping, mesh reconnection according to specified function
- Users have been able to start working with MSTK quickly
- Student was able to start experimenting with 3D mesh computations in a week
- Is quite robust - tested on medium size meshes (200,000 elements)

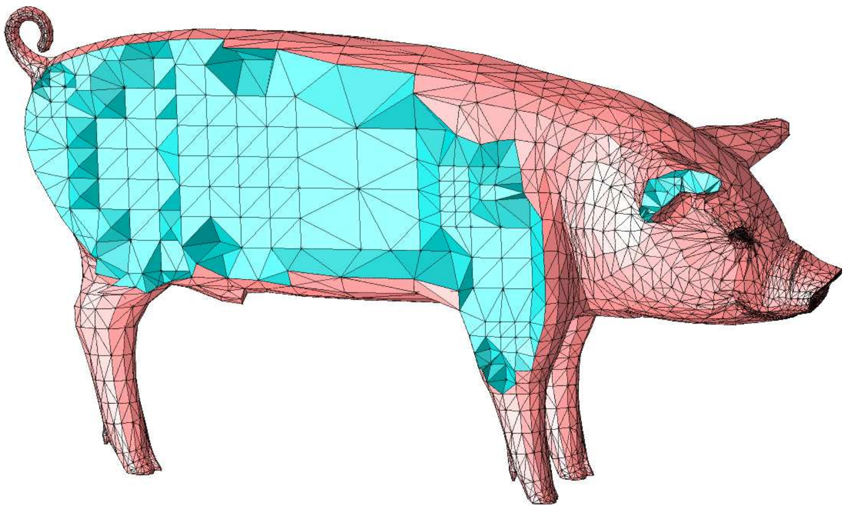
Application - Optimization of Polygonal Surface Meshes



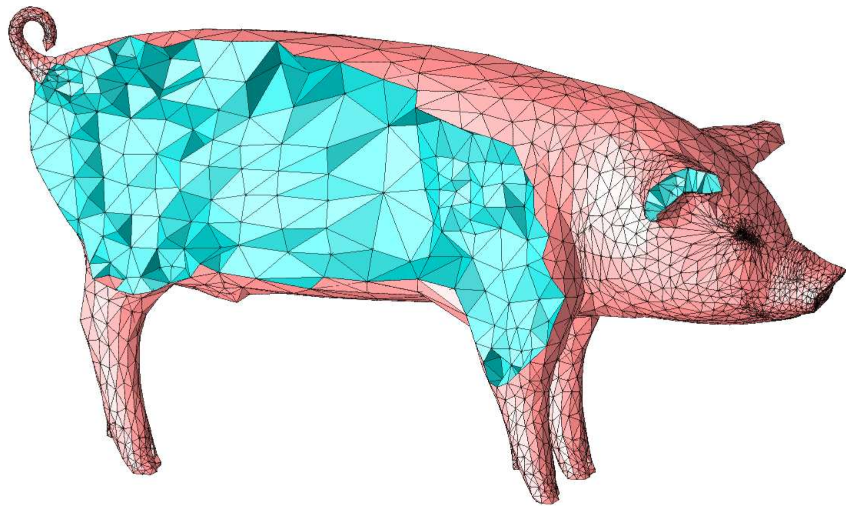


Original Mesh Optimized Mesh

Application - Optimization of 3D Tetrahedral Meshes



Original Mesh



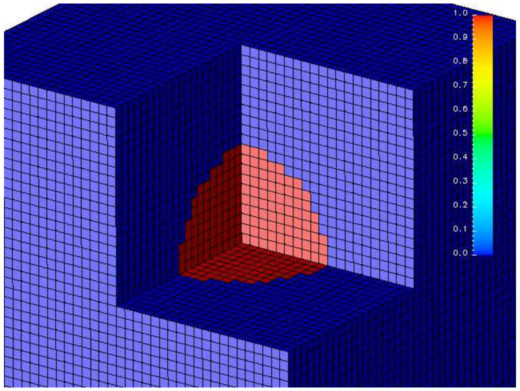
Optimized Mesh

Application - Local-Bound Preserving Remapping

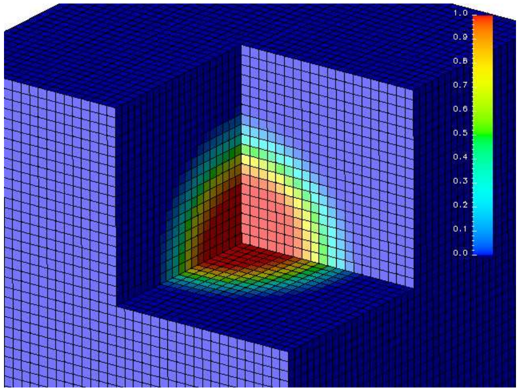
3D sphere function

$$f(x,y,z) = \begin{cases} 1 & \textbf{for } r \leq 0.25 \\ 0 & \textbf{else} \end{cases}$$

$$r = \sqrt{\left(x-\frac{1}{2}\right)^2 + \left(y-\frac{1}{2}\right)^2 + \left(z-\frac{1}{2}\right)^2}$$



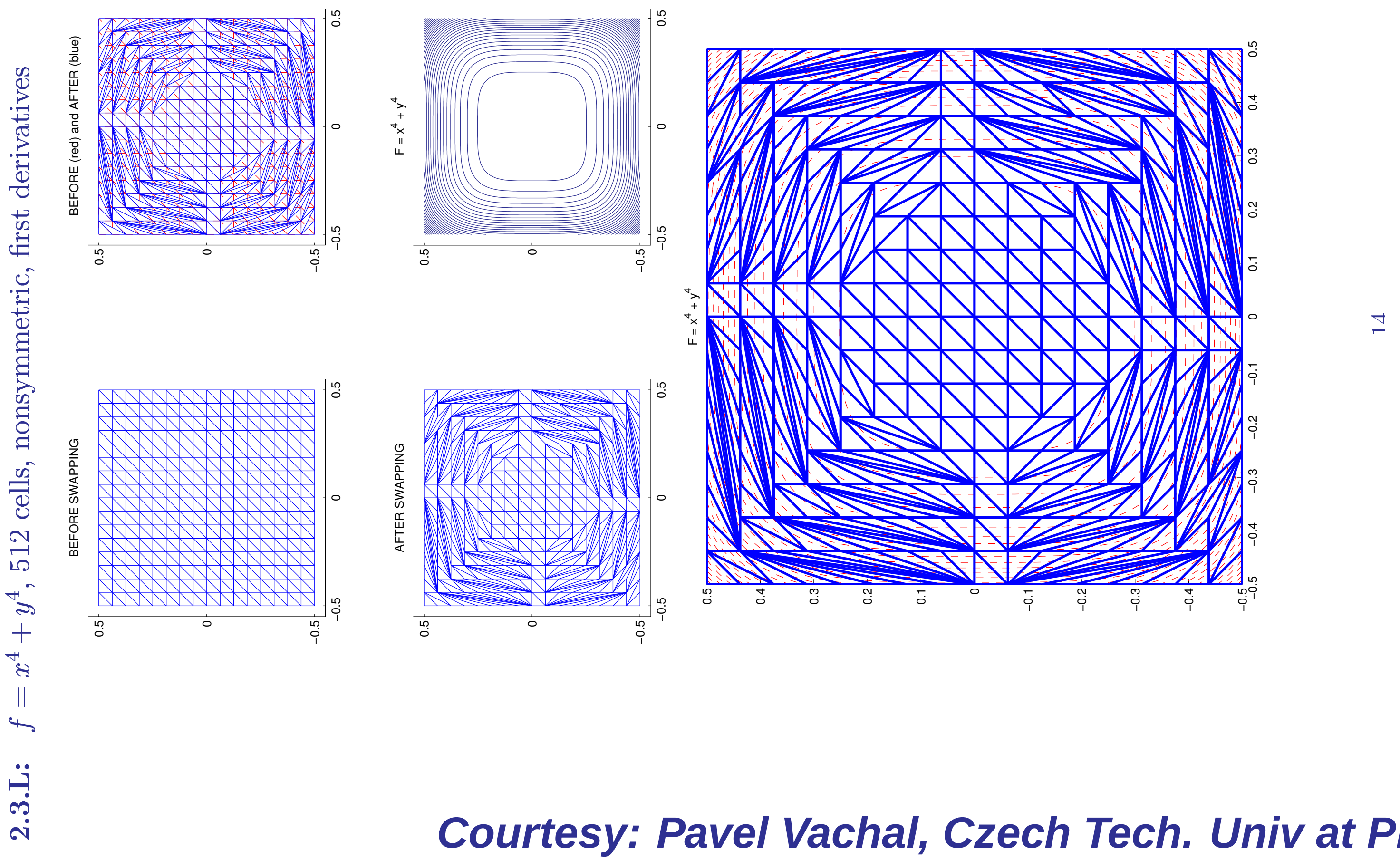
$40 \times 40 \times 40$ **grid, initial sphere**



$40 \times 40 \times 40$ **grid, after 200 remaps**

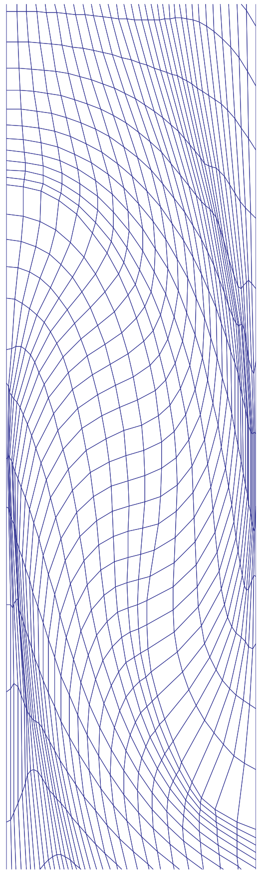
Courtesy: Milan Kucharik, Czech Tech. Univ at Prague

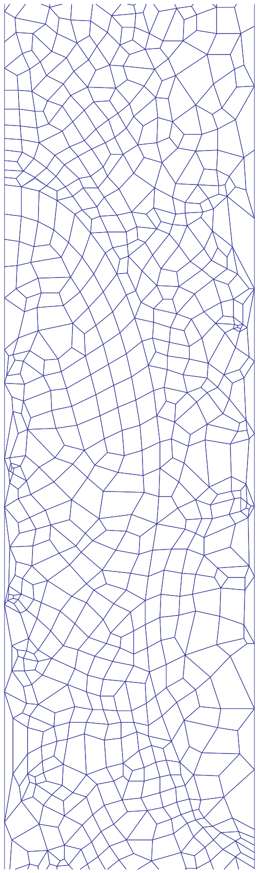
Application - Mesh Alignment



Courtesy: Pavel Vachal, Czech Tech. Univ at Prague

Application - “Reconnection” for Polygonal Meshes





Memory Requirements

- Statistics for tetrahedral meshes in representation F1
- Includes memory usage for entities and adjacencies
- About 400 bytes per element and 2000 bytes per node
- About 5.5 million elements on a 32-bit machine, 2 GB memory
- Many more elements with reduced representations

Reducing memory requirements of MSTK

- Investigating methods for reducing memory usage
- Spatial trees instead of linear arrays to store mesh entities
- Use local numbering for entities in each terminal node
- Reference entities by local numbers not pointers
- Use encoding techniques to compress integers storage
- Expect factor of 3 or more reduction in memory usage
- Will make use of unstructured meshes more viable

MSTK - Work in progress, Future Work

- Reduced representations with virtual edges, faces
- Dynamic switching between representations
- Association of field data with entities (scalar, vector)
- Wider range of mesh modification functions, 3D edge swap, edge refinement, templated region refinement
- Parallel MSTK for distributed meshes